Comparison of Existing Multipliers and Proposal of a New Design for Optimized Performance

R Dhanabal,V Bharathi,Anand N, George Joseph, Suwin Sam Oommen, Dr Sarat Kumar Sahoo
School of Electronics Engineering, VIT University, Vellore- 632014, TamilNadu, India
rdhanabal@vit.ac.in, bharathidhanabal@gmail.com

Abstract -Multipliers are inevitable components in digital system design and embedded applications. The performance parameters of multipliers play a vital role in maximizing the efficiency of these applications. Different algorithms have been proposed for improving the performance parameters of multipliers. This paper formulates a comparative study of some of the well known existing multipliers and thereafter proposes a robust design. The multiplier with a carry-look-ahead adder has shown a better performance over the multiplier with a ripple adder in terms of gate delays.

Keywords: **Binary array; urdhava tiryakbhyam; partial product; carry look ahead;**

Introduction

Multiplication finds a variety of applications in most of the Digital Signal Processing (DSP) applications. The speed of the multiplier is the impediment that determines the extent of pipelining in the ALU. It is in fact an indication of how quick a processor runs. The multiplication operation consists of producing partial products and then adding these partial products and the final product is obtained. Thus the speed of the multiplier depends on the number of partial product and the speed of the adder. Since the multipliers have a predominant effect on the system performance, many high performance algorithms and architectures have been proposed. It is evident that methodologies for the design of high-throughput, low-power digital systems are needed[1]. Today's technology has brought us to a stage where a new horizon of possibilities has opened up that the above paradoxical requirements no longer remain unfeasible.

We propose the redesign of the existing binary multipliers using ripple carry adder by modifying the last row of partial product generation of the

binary multiplier and using a carry look ahead adder. By using this approach we obtain an

efficient design in terms of delay optimization. On a comparative study with some of the existing multipliers, the proposed design achieves a better result[2],[3].The designs are implemented in Verilog HDL.

Existing multiplier designs

Binary array multiplier

The basic structure of an array multiplier consists of AND gates, full adders, and half adders. Here, a 4 bit parallel multiplier has been considered for analysis. This method of multiplication is called partial product accumulation because rows of linked adders generate the accumulated partial products that would evolve from a manual multiplication of the data words[4]. It can be seen that the lion's share of this array structure replicates its basic cell. A 4 bit combinational multiplier has 16 AND gates, 8 full adders and 4 half adders. We can generalize this by considering two numbers which are m and n bit wide respectively. There are m*n summands that are generated parallely by a set of m*n AND gates. Therefore an n x n multiplier requires 2*n AND gates, n half-adders and n*(n-2) full adders[5].

By using Array Multiplier, the number of components required is optimized, but delay for this multiplier is larger. It can be noted that the worst delay (ignoring routing delays) is independent of the path traversed in the array. Owing to the regularity of the structure, this path can be from LSB input to the product MSB. On a negative note, it can be said that due to the usage of large number of gates the hardware becomes complex and area is increased. Hence it is less economical[6].

Urdhava Tiryakbhyam (Vedic) multiplier

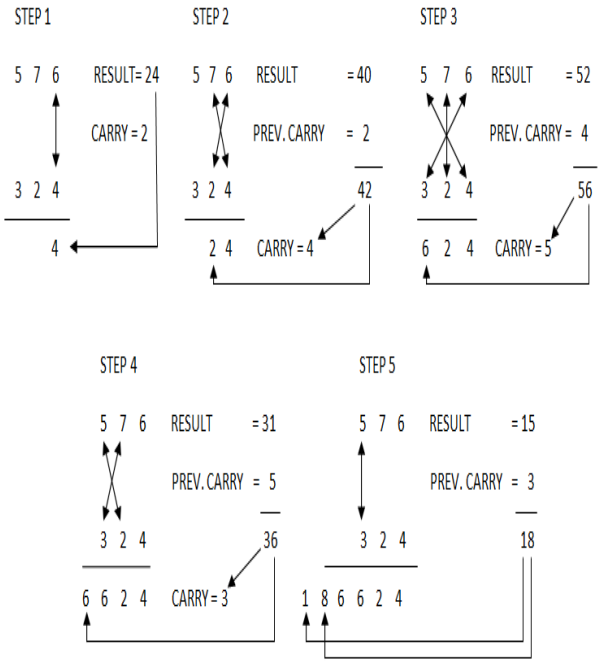

Fig. 1. Illustration of vedic multiplication

Vedic multiplication deals with 16 Vedic sutras. A wide range of mathematical problems can be effectively solved with the appropriate usage of these aphorisms. The fact that these methods are really fast when compared to the conventional mathematical methods has led to their application in processors to increase their computational speed. An added advantage is that they also require less hardware which can be attributed to the symmetric computation evident in this technique

The multiplier that is explained below is based on Urdhava Tiryakbhyam, one of the ancient sutras of the ancient Indian Vedic Mathematics. Urdhava Tiryakbhyam Sutra, which literally means “Vertically and crosswise”, is a general multiplication formula applicable to all cases of multiplication. This Sutra highlights parallelism in generation of partial products and their summation as depicted in Fig 1[7].

The algorithm can be generalized for n x n bit number. The concurrency while obtaining the partial products and their sums results in the non dependency of the multiplier on the clock frequency of the processor. The usage of this multiplier ascertains that microprocessors need not operate at increasingly high clock frequencies, thereby eliminating the chances of higher power consumption. In addition to this, the drawing of its layout is much easier because of the regularity in its structure. As compared to other multipliers, the gate delay and area increases gradually with the increase in the input bit size[8].

Proposed multiplier design

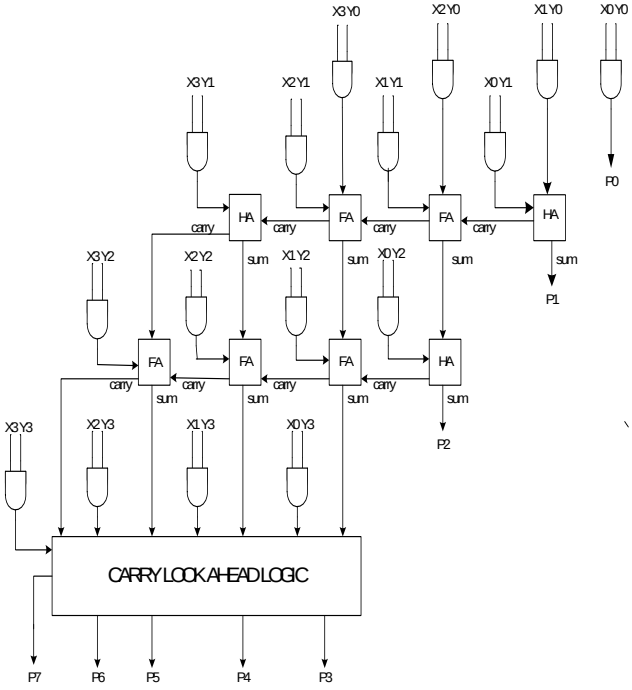

Fig. 2. Hardware for the proposed multiplier

The structure of a conventional array multiplier that multiplies two unsigned binary numbers has been developed from the manual approach for multiplying the numbers. It is an unaltered version of the standard shift-add multiplication algorithm. Beginning with the LSB of the multiplier, each bit is multiplied by the bits of the multiplicand to form a so called partial product. The operation that forms a partial product is logically equivalent to AND-ing each bit of the multiplier with the corresponding bit of the multiplicand. The partial products generated are shifted one by one towards the MSB of the multiplicand keeping it aligned with the position of the corresponding multiplier bit. Afterwards, the partial products are added. In the final step, we accumulate the result of adding the rows of the partial products, taking into consideration the carries which are generated by the addition of the terms in a particular column.

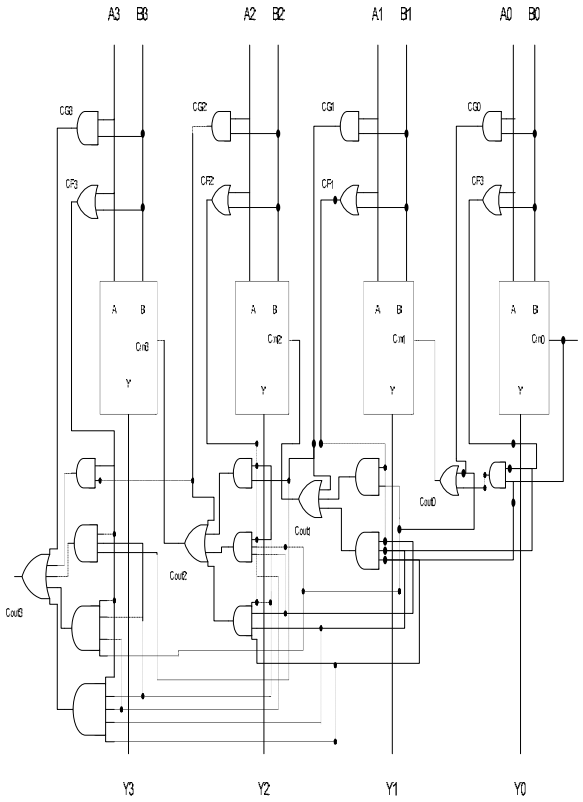

Fig. 3. Structure of carry look ahead adder[10]

It is obvious that we should use the fastest scheme since final addition time is a significant addendum to the critical path of the multiplier[9].The existing structure is modified by replacing the final full adder stage with a carry look ahead adder. For a parallel adder, the speed with which an addition can be realised depends upon the time taken for the propagation of the carries through all its stages. The carry look ahead adder accelerates the above process by getting rid of this ripple effect. All the input bits are simultaneously examined and the carry-in for each stage is generated. In order to decrease the delay produced by carry propagation through the ripple carry adder, the carry-in from previous stage will be evaluated quickly for each stage and a value of 0 or 1 is computed.

The sum and carry can be expressed as follows:

$SUM = A_i \oplus B_i \oplus C_i$

$C_{out} = C_{i+1} = A_i \cdot B_i + (A_i \oplus B_i) \cdot C_i$

If we let:

$G_i = A_i \cdot B_i$ -- The Generate Function

$P_i = (A_i \oplus B_i)$ -- The propagate Function

And we obtain:

$C_{i+1} = G_i + P_i \cdot C_i$ -- The Carry Function

The carry can be expressed as follows, for a four bit adder:

$$C_1 = G_0 + P_0 \cdot C_0$$

$$C_2 = G_1 + P_1 \cdot C_1 = G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_0$$

$$C_3 = G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot G_0 + P_2 \cdot P_1 \cdot P_0 \cdot C_0$$

$$C_4 = G_3 + P_3 \cdot G_2 + P_3 \cdot P_2 \cdot G_1 + P_3 \cdot P_2 \cdot P_1 \cdot G_0 + P_3 \cdot P_2 \cdot P_1 \cdot P_0 \cdot C_0$$

Taking into consideration each full adder stage, carry out depends on the initial carry-in, its generate function and propagate function. All the generate and propagate functions are expressed in terms of inputs to the full adders. The wait for the carry to ripple through every stage is now avoided since all the outputs are instantly available[10]. In effect the carry look ahead adder speeds up the addition process which results in a faster partial product generation in the proposed design.

Simulation results

The product ‘y’ is obtained for two 4 bit numbers ‘a’ and ‘b’

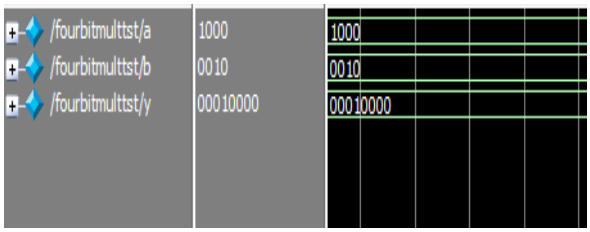

Fig. 4. Binary array multiplier

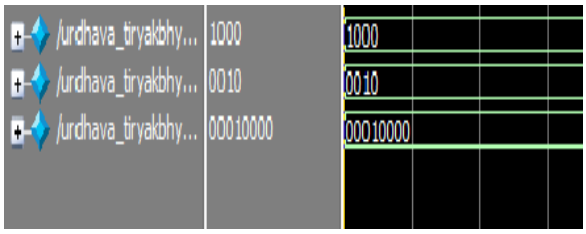

Fig. 5. Vedic multiplier

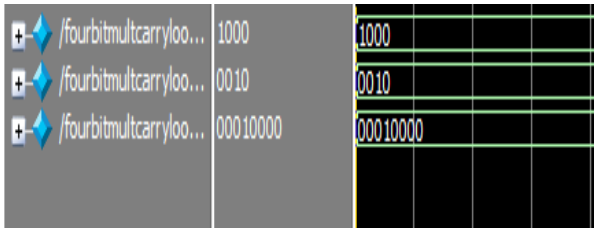

Fig. 6. Multiplier using carry look ahead

Performance comparison

TABLE I
Comparison of multipliers in terms of delay

TYPE OF MULTIPLIER	DELAY (ns)
Binary Array Multiplier	17.358
UrdhavaTiryakbhyam (Vedic) Multiplier	16.664
Proposed Multiplier with Carry Look Ahead Adder	16.620

TABLE II
Comparison of multipliers in terms of power

TYPE OF MULTIPLIER	POWER CONSUMPTION (mW)
Binary Array Multiplier	68.48
UrdhavaTiryakbhyam (Vedic) Multiplier	68.47
Proposed Multiplier with Carry Look Ahead Adder	68.49

Conclusion

This paper compares the aforesaid multiplier designs in terms of delay and power and thereafter proposes a new design using carry look ahead adder. Using the Classic Timing Analyzer tool, it was observed that the new multiplier design using carry look ahead adder has less delay compared to the other multiplier designs. The power consumption calculations of the multipliers were done successfully using Powerplay Power Analyser Tool in QuartusII mapping to the target device family CycloneII. It was observed that the power consumption of all the multiplier designs were more or less the same. Thus the paper comes up with a new multiplier design of less delay without compromising on the power consumption.

References

[1] Anantha P. Chandrakasan, Samuel Sheng, Robert W. Brodersen" Low-Power CMOS digital design," IEEE Journal of Solid State Circuits, Vol. 27, No. 4, 1992.
[2] Kelly Liew Suet Swee, Lo Hai Hiung, "Performance Comparison Review of 32-Bit Multiplier Designs," IEEE, 2011.
[3] Hasan Krad, Yousif Al-Taie Aws, "Performance Analysis of a 32-Bit Multiplier with a Carry-Look-Ahead Adder and a 32-bit Multiplier with a Ripple Adder using VHDL," Journal of Computer Science 4 (4): pp. 305-308, 2008.
[4] Michael D. Ciletti, "Advanced digital design with Verilog HDL," 2nd edition 662-667, 2009.
[5] Shaik Nasar, K. Subbarao, "Design & Implementation of MAC Unit Using Reversible Logic," International Journal of Engineering Research and Applications Vol. 2, Issue5, pp. 1848-1855, 2012.
[6] Sumit Vaidya, Dandekar. D, "Delay-power performance comparison of multipliers in VLSI circuit design," International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4, 2010.
[7] Swami Sri Bharati Krisna Jagadguru Tirthaji Maharaja, "Vedic Mathematics or Sixteen Simple Mathematical Formulae from the Veda, Delhi (1965), Motilal Banarsidas, Varanasi, India, 1986.
[8] Himanshu Thapliyal, Srinivas M. B, "An Efficient Method of Elliptic Curve Encryption Using Ancient Indian Vedic Mathematics," IEEE, 2005.
[9] Vojin G. O, D. Villeger, Simon S. L, "A Method for Speed Optimized Partial Product Reduction and Generation of Fast Parallel Multipliers Using an Algorithmic Approach," IEEE Transactions on computers, Vol. 45, No. 3, 1996.
[10] A. Anand Kumar, "Fundamentals of Digital Circuits," pp. 242-245 PHI Learning Private Ltd. New Delhi, 2008.